Data Leak Aware Crowdsourcing in Social Network

Iheb Ben Amor, Athman Bougetteya, Mourad Ouziri, Salima Benbernou, Mohamed Nadif

Data Leak Aware Crowdsourcing in Social Network

Iheb Ben Amor[1], Salima Benbernou[1], Mourad Ouziri[1], Mohamed Nadif[1], Athman Bouguettaya[2]

[1] Université Paris Sorbone Cité, Paris Descartes, France
[2] RMIT University, Australia
firstname.lastname@parisdescartes.fr
iheb.ben-amor@etu.parisdescartes.fr
athman.bouguettaya@rmit.edu.au

Abstract. Harnessing human computation for solving complex problems call spawns the issue of finding the unknown competitive group of solvers. In this paper, we propose an approach called *Friendlysourcing* to build up teams from social network answering a business call, all the while avoiding partial solution disclosure to competitive groups. The contributions of this paper include (i) a clustering based approach for discovering collaborative and competitive team in social network (ii) a Markov-chain based algorithm for discovering implicit interactions in the social network.

Keywords: Social network, outsourcing human-computation,privacy

1 Introduction

A new tend of teamwork has been emerged unconstrained by local geography, available skill set, networking and deep relationships the *crowdsouring*. It is the action of outsourcing tasks, traditionally performed by an employee or contractor, to an undefined group of people through an open call [1]. Crowdsourcing applications should be enable to seek for people crowd on demand to perform a wide range of complex and difficult tasks. Thousands human actors will provide their skills and capabilities in response to the call. We introduce a type of crowdsourcing called *Friendlysouring* based on the efficiency of social network to outsource a task to be performed by people on demand instead of an open world as Mechanical Turk is doing. In fact, the interactions between people involved to answer a query become complex more and more and the collaboration leads to the emergence of social relations and a social network can be weaved for human-task environment.

Challenges. The goal of Friendlysouring system is to let people collaborating on a joint task in the crowd environment where they may seek for other members towards social crowd relationships for achieving a business goal. Thus, many competitive teams can provide a set of answers to the call. However, as the crowd

task is competitive between teams, it is important to group people in a manner there is no inter-teams leaking. Such mechanism will avoid the information leak between crowd people in different teams. Hence, the issues and challenges considered in our system, include, (i) how to build up and discover teams answering the query towards the social relationships, (ii) how to avoid the solution disclosure of the problem during the teams construction between competitive groups. In fact, people on demand collaborating to a task may share sensitive information (part of the problem solution)that may be propagated or forwarded to other crowd members in the social network.

Few works dealing with crowdousourcing are provided. In [2], the Trivia Masster system generates a very large Database of facts in a variety of topics, cleans it towards a game mechanism and uses it for question answering. In [7] is proposed a novel approach for integrating human capabilities in crowd process flows. In [6], the CrowdDB system uses human input via crowdsourcing to process queries that neither database systems nor search engines can adequately answer. Privacy and data leaking are not at all discussed in these works. Moreover, privacy have been introduced in social network as in [3] to design a wizard that may automatically configure a user's privacy settings with minimal effort from the user to aim policy preferences learning. [8], the authors introduced privacy protection tool that measures the amount of sensitive information leakage in a user profile and suggest self-sanitizing action to regulate the amount of leakage. The primovoter tools is unable to estimate the leakage based on a private data propagation, so it settle for a direct user connections and an installed applications on friend profiles.

Contributions. We address the aforementioned challenges by proposing the Friendlysourcing system to discover data leaking aware competitive teams answering the query through a social network. The discovering method is based on a k-means like algorithm to cluster the potential crowd people from the social network that are *close* to collaborate in the same team. The system will not group them in the clusters that are competitive, thus, avoid inter-teams data leaking. To handle such leak, a Markov chain-based algorithm is proposed to discover the implicit social relationships between crowd people. It knows exactly to whom the user data can be propagated in the social network and hence avoid to let crowd people grouped in different clusters. The approach is based on dynamic model that deals with effective rates of shared data and not only on static friend relationship between crowd members.

The rest of the paper is organized as follows: section 2 provides an overview of our Friendlysourcing system. The propagation process is described in section 3. In section 4 is discussed the clustering based approach to discover competitive clusters in the social network answering a business call. Finally, conclusion and future works are given in the section 5.

2 Overview of Friendlysourcing Framework

We devise a crowdsourcing architecture for discovering data leak aware collaborative and competitive teams. It incorporates two main components discussed in this paper, they are depicted in Figure1 and are namely *Data propagation process, clustering process*. Beforehand, the person responsible of company call may register to the friendlysourcing platform using the user interface. He describes the company activities and submits a query. Once the call is launched, it will be visible in our platform. Every social network member is authorized to register for a call using the user interface. The registered member can examine the details of the call and make comments.

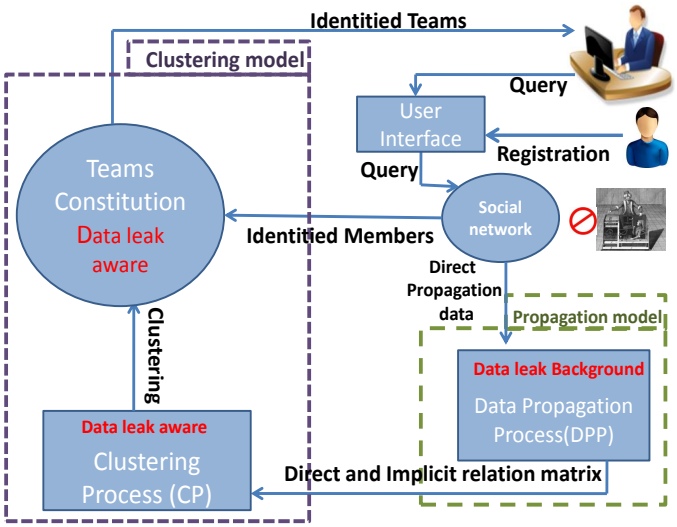

Fig. 1: Friendlysourcing architecture

- *Data Propagation Process :* When the registration is closed, the friendlysourcing system computes the data based on the information collected from the different social networks in order to data leak aware while discovering the teams. In the first step, the request is achieved by the propagation process, thus identifying the direct relationships between the social network members. After that, the process will discover implicit interactions in the social network and the maximum of data propagation between members. The approach is based on Markov chain model. The details are provided in the next section.
- *Clustering Process*
  In a second phase, starting from the whole data propagation calculated in the propagation process DPP, the Clustering process CP will group the crowd

users in the same clusters having strong propagation. That means, more the relationship probability is higher more the users need to be in the same cluster for the collaboration and not in competitive clusters.
- *Team Constitution*: The module will constitute the different team based on the provided result from the clustering process CP. It will use the user profile information provided from the social network. It will notify the users about the team discovering results.

3 A Markov chain-based approach for data propagation

The social networks is a set of direct relationships between members. These direct relationships allow to compute the probability of data propagation between only direct friends. However, discovering competitive teams aware of data leak, needs to know all possible interactions. For handling the implicit/indirect relations between members we propose a Markov chain-based approach.

We present in this section a model and an algorithm of data propagation across the entire social network. This allows to compute all indirect interactions between all members and to know to whom the user data can be propagated to.

3.1 A graph-based model of data sharing relationships

Our model of the social network is a labeled directed graph $G\langle M, A, P\rangle$ where,

- $M = \{m_i\}$: set of nodes where each node represents a member of social network.
- $A = \{a_{ij} = (m_i, m_j)/(m_i, m_j) \in M\}$: set of edges where each edge represents a direct friend relationship between two members.
- $P = \{p_{ij}/\forall i, j\ p_{ij} \in [0, 1]\}$: is set of labels where each label p_{ij} of the edge a_{ij} represents the rate/probability of data shared by the member $m_i \in M$ with his friend member $m_j \in M$.

In the given graph model, *friend* relationship is represented using edge A labeled with the real probability of shared data P.

The probability p_{ij} that the member m_i shares owned data with member m_j is computed in real time using the following formula:

$$p_{ij} = \frac{quantity\ of\ data\ that\ m_i\ shared\ with\ m_j}{quantity\ of\ data\ held\ by\ m_i}$$

Let's consider the example of a social network depicted in Figure 2:

- the arc (m_1, m_3) indicates that m_1 has friend relationship with m_3, and shares with him 90% of his data.
- the arc (m_1, m_2) indicates that m_1 has friend relationship with m_2 but he never shares with him any data.

The presented graph-based model is a set of direct relationships between members. These direct relationships provide the probability of data propagation between only direct friends. We present in the following the Markov propagation model to compute the probability of data propagation between indirect friends (such as propagation rate from m_1 to m_6 in Figure 2).

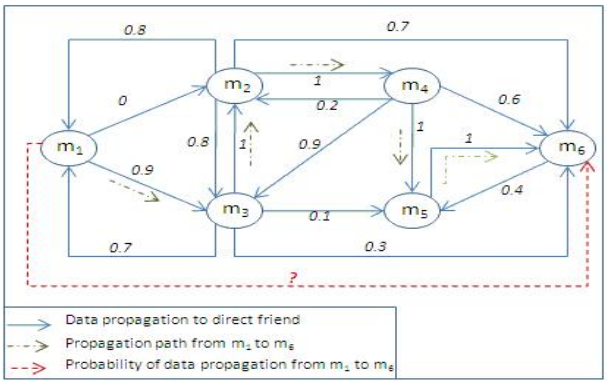

Fig. 2: Example of interactions between crowd members in the social networks

3.2 Markov chain-based model for data propagation

Given an owned data of a member, we propose a Markov chain-based model to compute the propagation probability of this data in the entire social newtork.

In social networks, data is propagated from friend to friend following a Markov chain model [4]. That is, a social network member shares owned data only with his friends and, then, each friend shares the data with only their friends and so on.

Definition 1. *A Markov chain is a sequence of random variables* $X_1, ..., X_n$ *with the Markov property, namely that, the future state depends only on the the present state, and not on the past states. Formally,*

$P(X_{n+1} = x | X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) = P(X_{n+1} = x | X_n = x_n)$

From this formal definition, the probability that a given member get a data depends on probability to get it from only his direct friends (and not from indirect friends).

The probability of data propagation between direct friends may be represented with Propagation Matrix defined as follows:

Definition 2. *Propagation Matrix of a social network is matrix that gives probability* p_{ij} *of propagating data between each couple of members* (m_i, m_j)*:*

$$\begin{array}{c} \\ m_1 \\ m_2 \\ m_3 \\ \ldots \\ m_n \end{array} \begin{array}{c} \begin{array}{ccccc} m_1 & m_2 & m_3 & \ldots & m_n \end{array} \\ \begin{pmatrix} p_{11} & p_{12} & p_{13} & \ldots & p_{1n} \\ p_{21} & p_{22} & p_{23} & \ldots & p_{2n} \\ p_{31} & p_{32} & p_{33} & \ldots & p_{3n} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ p_{n1} & p_{n2} & p_{n3} & \ldots & p_{nn} \end{pmatrix} \end{array}$$

where

$$p_{ij} = \begin{cases} \frac{quantity\ of\ data\ that\ m_i\ shared\ with\ m_j}{quantity\ of\ data\ held\ by\ m_i} & if\ (m_i, m_j) \in A \\ 1 & for\ i = j \\ 0 & else \end{cases}$$

This propagation matrix has the following properties:

- $p_{ii} = 1$, which means that member m_i does not lost owned data when he shares it.
- $\sum_{k\in[1,n]} (p_{ik}) \neq 1$, because data may be propagated to several members at the same time.
- $p_{ij} \neq p_{ji}$, which means that a member m_i may share with a friend m_j a quantity of data different his friend m_j may share with him.
- $\exists(i,j)|(m_i, m_j) \in A \wedge p_{ij} = 0$, which means that members do not share necessarily data with their friends.
- if m_i and m_j are not direct friends then $p_{ij} = 0$.

The propagation matrix of the social network of figure 2 is given as follows:

$$\begin{array}{c} \\ m_1 \\ m_2 \\ m_3 \\ m_4 \\ m_5 \\ m_6 \end{array} \begin{array}{c} \begin{array}{cccccc} m_1 & m_2 & m_3 & m_4 & m_5 & m_6 \end{array} \\ \begin{pmatrix} 1 & 0 & 0.9 & 0 & 0 & 0 \\ 0.8 & 1 & 0.8 & 1 & 0 & 0.7 \\ 0.7 & 1 & 1 & 0 & 0.1 & 0.3 \\ 0 & 0.2 & 0.9 & 1 & 1 & 0.6 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0.4 & 1 \end{pmatrix} \end{array}$$

3.3 A Markov chain-based algorithm of data propagation

The propagation matrix of section 3.1 gives only probability of data propagation between direct friends.

But it is not sufficient to compute the probability that data is propagated from member to indirect-friend because:

1. Propagation matrix defined in Definition 2 does not give the real propagation probabilities between members. In Figure 2, the direct propagation probability from m_1 to m_2 is zero ($p_{12} = 0$). However, through m_3, data of m_1 may be propagated to m_2 with probability $0.9 \times 1 = 0.9$.
2. The Propagation matrix does not provide the data propagation to indirect-friends. It's indicate a zero value of sharing data with indirect friends, because members share their data only with direct friends. In the propagation matrix of figure 2, probability that data of member m_1 may be propagated to his indirect-friends m_5 and m_6 is zero because m_1 is no direct friend relationship with them. However, data may be propagated from m_1 to m_5 through m_3 with probability $0.9 \times 0.1 = 0.09$.

3. Propagation to indirect-friends is hard to calculate: as example, what is the probability that data of m_1 may be propagated to m_6 (probability of dotted red arrow in figure 2)? To calculate this probability, we have to explore all the paths allowing propagation of data from m_1 to m_6. Each one allows to calculate a propagation probability. The final propagation probability is the maximum of propagation probability of all the possible paths, wihich corresponds to the propagation risk. The path $(m_1, m_3, m_2, m_4, m_5, m_6)$ indicated with dotted green arrows in figure 2 allows propagation of data from m_1 to m_6 with the maximum probability $0.9 \times 1 \times 1 \times 1 \times 1 = 0.9$. However, it is hard to calculate this probability because real social networks are complex.

For this reasons, we need to design an efficient algorithm that calculates the optimal data propagation probability from the owner to all the members of the social network. This algorithm is based on energy function we define as follows:

Definition 3. *The energy function p_i of member m_i is the probability that data is propagated to member m_i. In our model, data is propagated following Markov chain. That is:*

$$p_i = \underset{m_k \in N_{m_i}}{Max} (p_k \times p_{ki}) \tag{1}$$

where,

- N_{m_i} is a set of direct friends of m_i,
- p_k is the energy function of m_k,
- p_{ki} is the probability of propagating data from m_k to m_i.

To compute the energy function p_i for all members m_i of the social network, we have to use an iterative algorithm [9]. We design our simple algorithm Algorithm 1.

The algorithm processes as follows:

1. Initialisations: $p_{ow} = 1$, $\forall i \neq ow \; p_i = 0$. That is, only the owner m_{ow} has the data.
2. Iterations: At each iteration, the algorithm computes p_i for $m_i \in N_{m_i}$ using formula (1).
3. Stop: The algorithm stops when the probability maximum of each member is reached.

Applying this algorithm on the propagation matrix of section 3.2, we get the following matrix completed with indirect-friend propagation probabilities:

$$\begin{array}{c} \\ m_1 \\ m_2 \\ m_3 \\ m_4 \\ m_5 \\ m_6 \end{array} \begin{array}{c} \begin{array}{cccccc} m_1 & m_2 & m_3 & m_4 & m_5 & m_6 \end{array} \\ \begin{pmatrix} 1 & 0.9 & 0.9 & 0.9 & 0.9 & 0.9 \\ 0.8 & 1 & 0.8 & 1 & 1 & 1 \\ 0.8 & 1 & 1 & 1 & 1 & 1 \\ 0.7 & 0.9 & 0.9 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0.4 & 1 \end{pmatrix} \end{array}$$

Algorithm 1 PROPAGATION PROBABILITY COMPUTING ALGORITHM

Require: $G\langle M, A, PM\rangle$ – labeled directed graph of the social network where PM in the propagation matrix
m_{ow} – owner of the data
m_r – recipient member of data that we want calculate the propagation probability
Ensure: p_r – probability that owned data is propagated to m_r
1: **print** $\mathcal{P} = (p_1, \ldots, p_{ow}, \ldots, p_r, \ldots, p_n)$: Energy function at the previous step.
2: **print** $\mathcal{PS} = (ps_1, \ldots, ps_{ow}, \ldots, ps_r, \ldots, ps_n)$: Energy function at the current step.
3: **print** *continue*: boolean value indicating if the optimum values of all members are reached.
{I}nitializations
4: $p_{ow} = 1$ and $\forall m_i \neq m_{ow}, p_i = 0$
5: $continue \leftarrow true$
{I}terations
6: **while** *continue* **do**
7: **for** each members $m_i \neq m_{ow}$ **do**
8: $ps_i = Max_{m_k \in N_{m_i}}(p_k \times p_{ki}))$
9: **end for**
10: **if** $\mathcal{P} \neq \mathcal{PS}$ **then**
11: $\mathcal{P} \leftarrow \mathcal{PS}$
12: **else**
13: $continue \leftarrow false$
14: **end if**
15: **end while**
16: **return** p_r

4 Data disclosure aware clustering process

Based on data propagation calculated in the previous section, the clustering process groups of crowd members into free data leak clusters.

Definition 4. *A cluster C is set of crowd members having or no strong propagation:*

$$C = \{m_i\} \text{ such that } \forall m_i, m_j \in C, p_{ij} \in [0,1], p_{ji} \in [0,1]$$

Definition 5. *Two clusters C_k and C_s are free data leak iff:*

$$\forall m_i \in C_k, \forall m_j \in C_s, p_{ij} \leq \eta \wedge p_{ji} \leq \eta$$

From the definition 5, we consider there is a risk of data leak between two clusters C_k and C_s if the propagation rate between all the members of the two clusters is less than a threshold η. The later is proposed as a value for which the data propagation of a crowd member is acceptable in a social network. The dynamacity of social member profil impactes the value of η, but it is out of the scope of the paper.

The propagation matrix calculated by the algorithm 1 is be updated as follows:

$$\forall i, j, p_{ij} = Max(p_{ij}, p_{ji})$$

The propagation matrix of our example is updated as follows:

$$\begin{array}{c} \\ m_1 \\ m_2 \\ m_3 \\ m_4 \\ m_5 \\ m_6 \end{array} \begin{array}{c} \begin{array}{cccccc} m_1 & m_2 & m_3 & m_4 & m_5 & m_6 \end{array} \\ \begin{pmatrix} 1 & 0.9 & 0.9 & 0.9 & 0.9 & 0.9 \\ 0.9 & 1 & 1 & 1 & 1 & 1 \\ 0.9 & 1 & 1 & 1 & 1 & 1 \\ 0.9 & 1 & 1 & 1 & 1 & 1 \\ 0.9 & 0 & 1 & 1 & 1 & 1 \\ 0.9 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

Based on this updated propagation matrix, the members are classified into free leak clusters using a clustering algorithm.

Our clustering algorithm is a k-means algorithm [5]. The principle of the algorithm is that for each cluster C_k and each member m_i, the member m_i is calssified to the cluster C_k if there is high data propagation between the member m_i and at least one of the members of C_k. The algorithm uses the following specific distance called D_{max}:

$$D_{max}(C_k, m_j) = \underset{m_i \in C_k}{Max}\ p_{ij}$$

where Max is the maximum function, C_k a cluster to be built, m_j is member that can be clustered into the cluster C_k, and m_i is a crowd member in C_k, p_{ij} is the propagation value between m_i and m_j given in te propagation matrix.

The clustering algorithm prossess as follows:

- Inputs: the data disclosure threshold η, number of clusters.
- Initialization: The initialization of the clusters is done by assigning arbitrarily a member to each cluster.
- Iterations: For each candidate member m_i and a cluster C_j, if $D_{max}(C_j, m_i) \geq \eta$ then m_i is added to C_k
- Stop: The algorithm is deemed to have converged when the assignments of members to clusters no longer change.

Moreover, in the case that a candidate member has a strong communication with more than one cluster, we will merge the clusters with whom the candidate member has a high data propagation and assign it to the new merged cluster. Because it may probably disclosure the data of the cluster to the other clusters. For instance if $d(C_1, m_k) = 0,8$ and $d(C_2, m_k) = 0,7$, then we will merge the cluster C_1 and C_2 and K will integrate the cluster C_{12} result of the C_1 and C_2 fusion. The algorithm is presented as Algorithm 2.

5 Conclusion and future work

In this paper, we proposed a Friendlysourcing framework as a clustering based approach for data leak aware discovering competitive teams during the crowdsourcing process in social network. First, the Markov model is used to estimate

Algorithm 2 D-MAX DISCOVERING TEAMS ALGORITHM

print $Clusters = (Clust_1, Clust_2, ..., Clust_{Cluster})$: Teams constitution
{I}nitializations
$Distances, Centroid, MaxDistancesMax = 0$
$ClustNB = -1$
$Threshold = \eta$
for each $ClustersinCluster$ **do**
  $Clusters_i = member_{m_i}$
  $Distances_i = 0$
  $Centroid_i = 0$
end for
{I}terations
for each members m_i in G **do**
  for each members m_i in $Clusters$ **do**
    $Distances_i \leftarrow P_{member,member_i}$
    if $Distances_i \succ Centroid_i$ **then**
      $Centroid_i \leftarrow Distances_i$
    end if
  end for
  for each $Centroid_i$ **do**
    if $Centroid_i \succ MaxDistancesMax_i$ **then**
      $MaxDistancesMax_i \leftarrow Centroid_i$
      if $MaxDistancesMax_i \succ \eta$ AND $ClustNB \neq -1$ **then**
        $Clusters_i = FUSION(Clusters_i, Clusters_{NB})$
      end if
      $ClustNB = i$
      $Clusters_{ClustNB} \leftarrow member_i$
    end if
  end for
end for
return $Clusters$

the hidden relationships between crowd members in the social network. Given the results of the previous step, then, the clustering approach groups the crowd members into data leak aware competitive teams.

In the future works, we plan to evaluate efficiency of the proposed approach by means of data leakage and time consumption. Regarding the social network complexity, the current approach provides several classifications of competitive teams but not easy to choose the best one. Then, we will study how to take into consideration more constraints specifically user preferences.

References

1. Daren C. Brabham. Crowdsourcing as a model for problem solving: An introduction and cases. *Convergence: The International Journal of Research into New Media Technologies*, 14(1):75–90, 2008.

2. Daniel Deutch, Ohad Greenshpan, Boris Kostenko, and Tova Milo. Using markov chain monte carlo to play trivia. In *ICDE*, pages 1308–1311, 2011.
3. Lujun Fang, Heedo Kim, Kristen LeFevre, and Aaron Tami. A privacy recommendation wizard for users of social networking sites. In *ACM Conference on Computer and Communications Security*, pages 630–632, 2010.
4. Olle Hggstrm. Finite markov chains and algorithmic applications. In *in London Mathematical Society Student Texts*. Cambridge University Press, 2000.
5. J. MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, page 14. California, USA, 1967.
6. Timos K. Sellis, Renée J. Miller, Anastasios Kementsietsidis, and Yannis Velegrakis, editors. *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2011, Athens, Greece, June 12-16, 2011*. ACM, 2011.
7. Florian Skopik, Daniel Schall, Harald Psaier, Martin Treiber, and Schahram Dustdar. Towards social crowd environments using service-oriented architectures. *it - Information Technology*, 53(3):108–116, 2011.
8. Nilothpal Talukder, Mourad Ouzzani, Ahmed K. Elmagarmid, Hazem Elmeleegy, and Mohamed Yakout. Privometer: Privacy protection in social networks. In *ICDE Workshops*, pages 266–269, 2010.
9. Thomas Weise. *Global Optimization Algorithms - Theory and Application*. June 2009.